Security Analysis of the Secure Authentication Protocol by Means of Coloured Petri Nets

Wiebke Dresp

Department of Business Information Systems,
University of Regensburg
wiebke.dresp@arcor.de

Abstract. Wireless communication demands for specialized protocols secure against attacks on the radio path while fitting the limited calculation and memory capabilities of mobile terminals. To ensure accessibility of mobile services beyond a user's home network, signing on a foreign network should be possible. The latter must be able to authenticate a user without learning any secret registration data. Chouinard et al. [DBC01] introduce the Secure Authentication Protocol for this purpose.
In this paper, an exhaustive security analysis of the protocol is presented. First, it is mapped to a coloured petri net. Then, two different intruder models are developed and integrated separately into it. The state spaces of the two nets are calculated; they each contain a set of nodes representing all reachable states. Both are examined to detect states where any security objective is violated indicating a security flaw in the protocol. As there are no such states in both nets, the protocol is proven secure.

Keywords: Secure Authentication Protocol, Coloured Petri Nets, Formal Protocol Verification, State Space Analysis, Security Analysis.

1 Security in Wireless Communication Networks

To gain access to mobile communication services such as telephony or data transfer, users (or their mobile terminals, respectively) have to be registered at a service provider called a user's home agent. He represents the union of a network infrastructure and a registration database where the users' data including authentication data is stored. It is to be kept secret for privacy reasons.
User and home agent usually communicate via the radio path. As radio waves spread out into all directions, all radio receivers within transmission range can obtain the exchanged data and it is easy to send spurious data to the communicating entities as well. Thus communication over the radio path has to be secured by cryptographic techniques. With respect to limited calculation and memory capabilities of mobile terminals, use of public key cryptography has to be cut down to an absolute minimum. The low bandwith of the radio path has also to be taken into account for appropriate protocol design.

Due to terminal mobility it is probable that a user leaves the range of his home agent making direct communication infeasible. To have nonetheless access

J. Dittmann, S. Katzenbeisser, and A. Uhl (Eds.): CMS 2005, LNCS 3677, pp. 230–239, 2005.

to mobile services, it should be possible for such a user to contact a network (called foreign agent) available at his current location. This scenario of logging on a foreign network is widely called roaming.
At first, the foreign agent does not have any information about the user but, to prevent fraud, needs to find out if the user is authentic. Only if he is sure about this, he is willing to provide services with cost. The authenticity of a user can be confirmed only by his home agent; on that account, the foreign agent has to contact him and ask for authentication on the basis of data provided by the user. Only if the response is positive, the foreign agent approves the user's logon. Note that each user only trusts his home agent and will never submit any of his secret registration data to a foreign agent. Therefore, the foreign agent has to trust the home agent that his answers are right.

A relevant task in protocol design is assuring the privacy of the registration data between the user and his home agent while communication between them can only be realized with the foreign agent as intermediate.

2 Secure Authentication Protocol

2.1 Entities and Security Objectives

The Secure Authentication Protocol presented in [DBC01] is tailored to the security demands of the entities participating in a roaming situation. Three regular entities with different security objectives participate:

- **User *A* (Alice)** wants to be sure that she is properly informed about HA's answer. Note that person and mobile terminal form a combined entity.
- ***HA* (Home Agent)** holding A's registration data; there is a strong trust relationship between A and HA. HA wants to be sure that the authentication request with A's data was genuinely generated and sent by A.
- ***FA* (Foreign Agent)** in proximity to the mobile handset. FA does not know any of A's data and there is no trust relationship between these two entities. FA wants to be sure that A is properly authenticated since he wants to limit the risk of being bilked of the invoice for services requested by A. This objective is met if FA receives a positive answer by HA if A has submitted correct data and a negative one otherwise.

2.2 Protocol

Symmetric and asymmetric encryption and decryption techniques referred to as $E_{asymm}(x; pk_X)$, $D_{asymm}(y; sk_X)$ and $E_{symm}(x; sesk_1)$, $D_{symm}(y; sesk_1)$ with plain text x and cipher text y are used with pairs of public and secret[1] keys (denoted pk_X, sk_X for entity X) and symmetric keys (serially numbered $sesk_1, ...$). H names a collision resistant hash function. $S(x, sk_X)$ and $V(y, pk_X)$ form a corresponding pair of signature and verification functions.

[1] To keep abbreviations distinguishable, the term "private key" is not used in this paper.

There is a certificate-based trust relationship between FA and HA[2]. A secure channel can thus be established between them based on their public keys. To simplify matters, messages x sent via this channel are denoted $SecChannel(x)$ and in the following treated as plain text. A and HA share a secret password pwd_A as part of A's registration data. A protocol run[3] is carried out as follows:

1. FA sends a broadcast $broadc_{FA} = (pk_{FA}, loc_{FA})$ for all users in the transmission range. With FA's public key pk_{FA}, everybody can encrypt data for FA. loc_{FA} is some location information.
2. A creates a session key $sesk_1$ and calculates $encsesk_A = E_{asymm}(sesk_1; pk_{FA})$. Then she encrypts her request:

   $$encreq_A = E_{symm}((id_A, pk_A, n_1, dp_A, hv_A); sesk_1)$$

   with nonce n_1, A's mobile device profile dp_A and a hash value hv_A built with HA's domain name dn_{HA_A}[4]: $hv_A = H(id_A, n_1, dn_{HA_A}, dp_A, pk_A, pwd_A)$.
   A sends $encsesk_A$ and $encreq_A$ to FA.
3. FA subsequently calculates $sesk_1 = D_{asymm}(encsesk_A; sk_{FA})$ and from that $D_{symm}(encreq_A; sesk_1)$. The resulting values id_A, pk_A, n_1, dp_A and $sesk_1$ are stored. FA derives dn_{HA_A} from id_A and establishes a secure channel with HA. He submits $authreqfw = SecChannel(id_A, pk_A, n_1, dp_A, hv_A)$.
4. HA checks

   $$hv_A \stackrel{?}{=} H(id_A, n_1, dn_{HA}, dp_A, pk_A, pwd_{A_{HA}}) = hv_{HA}.$$

   HA takes dn_{HA} and $pwd_{A_{HA}}$ from the registration database. It is also checked if n_1 was used in a previous run in order to prevent replay attacks. In case of $hv_A = hv_{HA}$, HA sends a positive authentication response

   $$authresp_{ACK} = SecChannel(id_A, ACK, hv_{HA}, sesk_2, n_2, n_3, cert_A)$$

   to FA. ACK is a string of acknowledgement which possibly contains further details, e.g. the generation time of the response. The nonces n_2 and n_3 will later be used by A to generate session keys. $sesk_2 = H(n_1, n_2, pwd_A)$ is included so that FA can communicate securely with A without reusing $sesk_1$. HA also calculates and stores $sesk_3 = H(n_1, n_3, pwd_A)$. The certificate $cert_A = S((id_A, pk_A); sk_{HA})$ is submitted to A so that she can authenticate to other mobile terminals and service providers without involving HA in the future. Note that the certificate is modeled in a very simple way as there is no more detail given in [DBC01].
   In case of $hv_A \neq hv_{HA}$ or replayed n_1, HA's authentication response is

   $$authresp_{NACK} = SecChannel(id_A, NACK, hv_A, hv_2, n_2)$$

[2] Each entity is sure about the authentity of the public key belonging to the couterpart's identity due to a certificate issued by a trusted third party.
[3] taken from [DBC01] with correction of a misprint discussed in personal correspondence with the authors
[4] Due to the format $userX@domainHA$, dn_{HA_A} can also be derived from A's ID.

with $hv_2 = H(pwd_A, n_2, hv_A)$. hv_2 proves that the negative authentication response was actually generated by HA. $NACK$ is a string with information on the rejection.

5. FA submits to A

$$authrespfw_{ACK} = E_{symm}((id_A, ACK, hv_{HA}, n_3, cert_A); sesk_2), n_2$$

or

$$authrespfw_{NACK} = E_{symm}((id_A, NACK, hv_A, hv_2, n_2); sesk_1), n_1.$$

6. A calculates $sesk_2 = H(n_1, n_2, pwd_A)$. If she has received a positive response, the message can be decrypted with it. id_A and hv_{HA} are checked; in case of a positive outcome, $cert_A$ is stored[5] and $sesk_3 = H(n_1, n_3, pwd_A)$ calculated. If A has received a negative answer $authrespfw_{NACK}$, she can decrypt it with $sesk_1$ and compares id_A and hv_A to her stored values. She also checks $hv_{2_A} = (pwd_A, n_2, hv_A) \stackrel{?}{=} hv_2$. If this is the case, she accepts the rejection.

3 Modeling the Protocol

In protocol analysis, cryptography is treated as a secure black box. This means it is assumed an intruder cannot compromise any cryptographic technique.

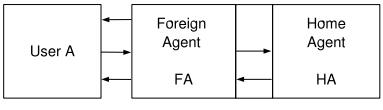

Fig. 1. Entities and Communication Paths in the Secure Authentication Protocol

3.1 Coloured Petri Net

Coloured petri nets [Jen92] have already proven suitable as a modeling technique for analysis of cryptographic protocols [DTM95] [DTM96] [Dre04]. They follow an elaborated mathematical syntax and provide a clear, intuitive and demonstrative graphical representation of the model thus facilitating its simulation and analysis which is a basic strength compared to other verification methods.

Data is modeled by tokens each belonging to a special data type called the colour set of a token. The token colour is the actual assignment of values to this token[6]. Figures 2 and 3 show the coloured petri net model of the Secure Authentication Protocol.

[5] Note that A does not verify the certificate herself.

[6] There is an analogy with object-oriented programming languages where objects carry certain attributes with attribute values.

3.2 Intruder Models

Following the intruder model of Dolev and Yao [DoY81], the intruder has to be modeled with the highest imaginable strength so that *all* possible attacks on the protocol can be identified. Considering the radio path insecure, the intruder has full control over it. According to the model, he can then carry out the following actions:

- Tapping and storage of all messages exchanged via the radio path
- Forwarding, rerouting and blocking of messages
- Generation of forged messages using tapped, randomly generated and obsolete data and encryption techniques
- Decryption of ciphertext if the intruder has a matching key

There are two different intruder models conceivable for this protocol:

- An intruder can try to intervene on the radio path and thus deceive all regular entities.

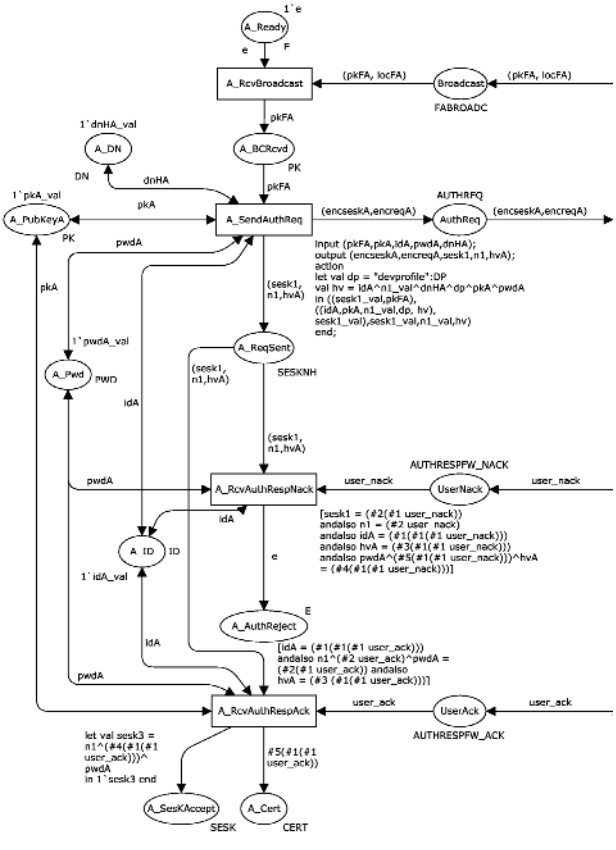

Fig. 2. Petri Net Model of the Secure Authentication Protocol (Entity *A*, Radio Path)

– Since A does not trust any entity except HA, it must as well be considered that FA might be malicious.

The second case considers a much stronger intruder. But as FA is not an honest protocol participant in that case, the compliance of *his* security objectives should be checked in the first model only. In both cases, the intruder may conspire with a registered user.

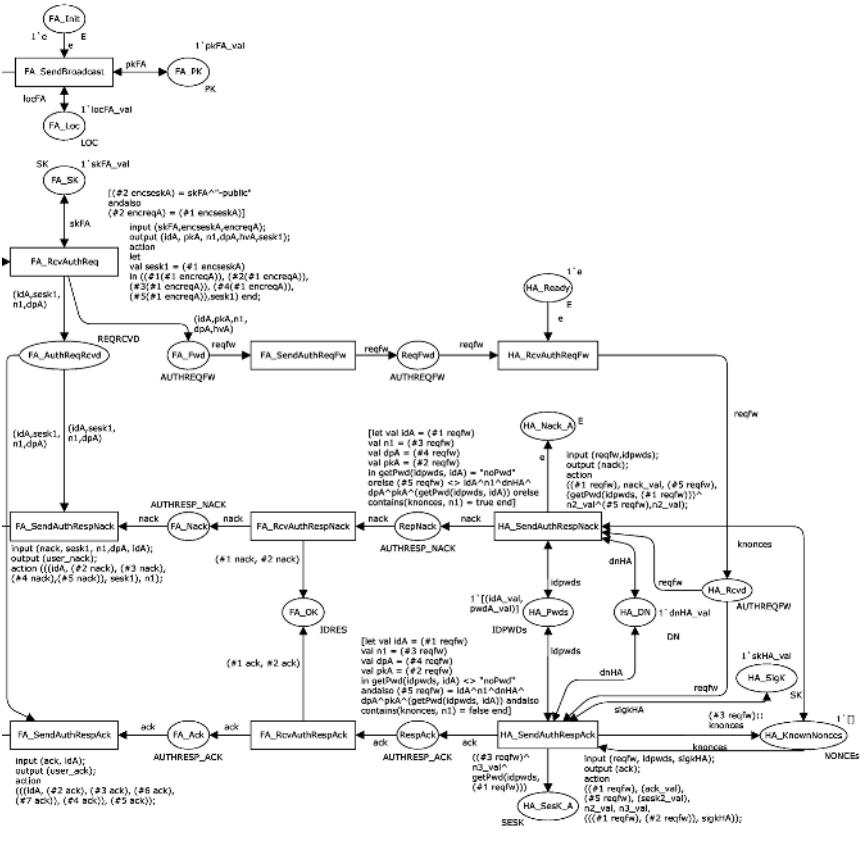

Fig. 3. Petri Net Model of the Secure Authentication Protocol (Entities FA, HA)

To perform an exhaustive security analysis, two different coloured petri nets have to be modeled, each including one of the identified intruders.

Intruder on the Radio Path. This model assumes an intruder on the radio path indicating FA is reliable. He can thus act as a foreign agent to A and as a user to FA or collaborate with a user cheating only FA. The intruder is integrated into the petri net as presented in figure 4.

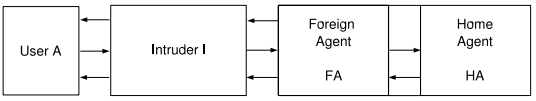

Fig. 4. Model with Intruder on the Radio Path

The entity mirrors A's and FA's sending and receiving transitions. Further, it is equipped with

- Places for storing all tokens received or generated during the protocol run, i.e. places for each colour set used by either A or FA
- Transitions for generation of requests, responses and their parts
- Transitions for decryption of all messages exchanged between FA and A

Malicious Foreign Agent. The foreign agent entity presented in figure 3 is extended to form an intruder (see figure 5). He can collaborate with a user to cheat HA or act on his own deceiving both A and HA. The most important modifications are:

- Places for each colour set used by HA or A
- Transitions for generation of authentication responses
- Transition for generation of spurious certificates
- Transitions for assembling spurious forwarded authentication requests

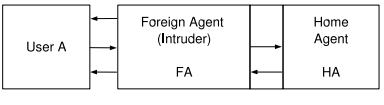

Fig. 5. Model with a Malicious Foreign Agent

4 Security Analysis

4.1 Relevant States in State Space

The state of a net is the assignment of all places with tokens. It changes when the number, positions, types and / or colours of tokens in the net are modified by transition firing. States are nodes in a digraph connected with edges each representing the firing of a binding element, i.e. a pair (t, b) with transition t and binding b (the assignment of token colours to each of the transition's variables). The *state space* represents the states reachable from the initial marking as a result of every possible permutation of transition firings. It is thus the exhaustive enumeration of all potential states and can be used to examine the security of a protocol as proposed in [DTM95], [DTM96] and [Dre04]. If any successful attack

can be carried out, there has to be at least one state where any security objective is violated, and there is a path of fired binding elements from the initial state to the respective state describing the attack. If no such states exist, it is proven that the protocol is in fact secure concerning the specified objectives.

For the Secure Authentication Protocol, states violating the security objectives described in 2.1 are:

1. Incorrect *user_ack*: HA has built and sent a positive answer though A did not initiate the authentication process. These are all states with a token in HA_SesK_A (i.e. HA generated a positive response) but with a token in A_Ready or A_BCRcvd implying A did not send any request.
2. Incorrect certificate: I holds a certificate legally signed with HA's secret key. It contains id_A bound to a public key different from pk_A.
3. Compromised *sesk_3*: I has found out the $sesk_3$. These are all states where I holds a token with the same colour as the token in HA_SesK_A.
4. Forged *user_ack* or *user_nack*: A has accepted a positive or negative response although HA did not send the respective message. This includes scenarios where I has generated a response without involvement of HA as well as those where I has intercepted HA's response and generated a converse message.

The security objectives of FA are contained implicitly in 1. As message integrity between HA and FA cannot be attacked due to the secure channel, FA can be cheated only if an intruder makes HA produce wrong responses.

4.2 Determination of the Optimal Initial Marking

The initial knowledge of the intruder has to be chosen carefully. Due to the state explosion problem [Val98], an initial marking containing too many tokens can cause problems regarding the computability of the net's state space. It has, on the other hand, to be avoided that the initial marking is improperly small so that tokens allowing for successful attacks are missing. In this case, the state space analysis might claim security of a protocol which is *not* secure against *all* possible attacks.

To determine the optimal initial marking, the maximum number of tokens to be used in *one* protocol run is identified and then reduced with a semi-formal rationale; multiple runs will be discussed later in this paper.
First, random tokens generated by the intruder himself are addressed. This number can be derived from examination of the transitions for sending and encrypting messages. Note that only those tokens that cannot help to perform a successful attack - e.g. because they can never pass a specified check by a regular entity - are sorted out in the reduction steps. The following principles apply:

- An intruder can send out broadcasts needing a key pair and a true location information for this purpose.
- An intruder cannot guess any secret key to sign a certificate that could be accepted by any other entity[7].

[7] Although A does not verify a received $cert_A$ herself, she will use it communicating with others who surely will.

- Messages generated by means of public key cryptography always have to be created with the public key of the receiver. Random public keys and encrypted messages cannot be used.
- No intruder can guess a password or retrieve it otherwise as it is sent by A or HA only after applying the one-way hash function. Accordingly, he cannot generate hash values as this requires knowledge of a password.
- The session keys $sesk_2$ and $sesk_3$ are calculated using a password and can hence not be generated by then intruder. Session key $sesk_1$ is chosen by A and could also be made up by the intruder, but this is not necessary: he will receive a session key from A after sending out a public key broadcast. Random encrypted values cannot be used either as they lack a matching session key.
- Random IDs lead to rejection as HA only accepts requests with a known ID and a matching hash value (which I cannot create). Device profiles and domain names can also be useful only if they match a hash value which does not hold for random tokens.
- Nonces have to match a certain hash value or session key. This claim cannot be met as the intruder cannot generate valid hash values.

Thus, the initial marking concerning random tokens is reduced to pk_I, sk_I, loc_I for both intruder models.

As stated in 3.2, the intruder could also plot with a regular user X[8] to login as A and receive a certificate $cert_{A_I} = E_{asymm}((id_A, pk_X); sk_{HA})$. But as HA always calculates hv_{HA} depending on the password pwd_A associated with id_A in the database, he will not accept requests built with pwd_X.

Now, multiple protocol runs are considered: they can be simulated by equipping the model with data collected in previous protocol runs. It has to be determined if the intruder can use such old messages anywhere. As a nonce is included in each request and checked by HA, old requests will be detected leading to a negative authentication answer. Old authentication answers are useless as they can only be decrypted by A if generated with $sesk_1$ (chosen by A in the current protocol run) or $sesk_2 = H(n_1, n_2, pwd_A)$ (with n_1 chosen by A in the current run). This leads to the conclusion that no old tokens can be used for a successful attack on the protocol i.e. that multiple protocol runs cannot lead to better attack results.

4.3 State Space Analysis Results

The models were generated with the *CPN Tools* software developed at the University of Aarhus [CPN]. Its State Space Tool was used for calculation and analysis of the state spaces. Table 1 shows the results: there are no illegal states according to 4.1.

Since both intruders cannot succeed in compromising the security objectives, **the petri net verification technique attests that the Secure Authentication Protocol does not possess any protocol flaws**.

[8] X is registered at HA and that HA therefore contains a pair of ID and password (id_X, pwd_X) in his database.

Table 1. State Space Details

Intruder Model	Initial Marking	State Space Size	Illegal States
Intruder on radio path	pk_I, sk_I, loc_I	523	0
Malicious foreign agent	pk_I, sk_I, loc_I	136	0

5 Conclusion

A coloured petri net modeling the Secure Authentication Protocol has been presented. Two different intruder models have been integrated separately. The optimal initial markings have been derived to cope with the state explosion problem without influencing completeness and accuracy of the analysis. Evaluation of the nets' state spaces has shown the absence of illegal states so that the Secure Authentication Protocol is evidently secure against attacks on the protocol level.

References

[CPN] CPN Tools Homepage: http://wiki.daimi.au.dk/cpntools/cpntools.wiki

[DBC01] Dupré la Tour I., Bochmann G. v., Chouinard J.-Y.: A Secure Authentication Infrastructure for Mobile Communication Services over the Internet. Proceedings IFIP Working Conference CMS'01, 2001, pp. 405 - 416.

[DoY81] Dolev D., Yao A.: On the Security of Public Key Protocols. Proceedings IEEE Symposium on Foundations of Computer Science, 1981, pp. 350 - 357.

[Dre04] Dresp W.: Computer-gestützte Analyse von kryptographischen Protokollen mittels gefärbter Petrinetze. Diploma Thesis, Department of Business Information Systems, University of Regensburg, 2004.

[DTM95] Doyle E., Tavares S., Meijer H.: Automated Security Analysis of Cryptographic Protocols Using Coloured Petri Net Specifications. Workshop on Selected Areas in Cryptography, SAC '95 Workshop Record, 1995, pp. 35-48.

[DTM96] Doyle E., Tavares S., Meijer H.: Computer Analysis of Cryptographic Protocols Using Coloured Petri Nets. 18th Biennial Symposium on Communication, Kingston, Ontario, 1996, pp. 194-199.

[Jen92] Jensen K.: Coloured Petri nets. Basic concepts, analysis methods and practical use, Vol. 1. Monographs in Theoretical Computer Science, Springer, 1992.

[Val98] Valmari A.: The State Explosion Problem. Lecture Notes in Computer Science, Vol. 1491: Lectures on Petri Nets I: Basic Models, Springer, 1998, pp. 429-528.